



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/028,363	12/21/2001	Dean T. Yu	04860.P0945C2	4131

7590 03/26/2004

James C. Scheller, Jr.  
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP  
Seventh Floor  
12400 Wilshire Boulevard  
Los Angeles, CA 90025-1026

EXAMINER

BANANKHAH, MAJID A

ART UNIT	PAPER NUMBER
----------	--------------

2127

DATE MAILED: 03/26/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

# Office Action Summary

Application No.

10/028,363

Applicant(s)

YU ET AL.

Examiner

Majid A Banankhah

Art Unit

2127

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

## Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

## Status

- 1) ☒ Responsive to communication(s) filed on 21 December 2001.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

## Disposition of Claims

- 4) ☒ Claim(s) 1-20 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 1-20 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

## Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

## Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
- ☐ Certified copies of the priority documents have been received.
  - ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  - ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

## Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☒ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)  
Paper No(s)/Mail Date \_\_\_\_\_.
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date. \_\_\_\_\_.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: \_\_\_\_\_.

## DETAILED ACTION

1. This office action is in response to application filed on December 21, 2001. Claims 1-20 are considered for examination.

### Obviousness-type double patenting Rejection:

2. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. See In re Goodman, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); In re Longi, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); In re Van Ornum, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); In re Vogel, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and In re Thorington, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

"Double patenting rejection of application claims was fully justified where applicant, in course of expanding first application to disclose enough more by way of details, alternatives, and additional uses to support broad, dominating, generic claims in later applications, has disclosed no additional invention or discovery other than that what was already claimed in patent on first application; there is significant difference between justifying broadening of claims and disclosing additional inventions." In re Van Ornum, 214 USPQ 761 (CCPA 1982).

Claims 1-20 are rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over claim 1-19 of application 08/558,929, now U.S. Patent No. 6,430,685 (hereinafter '685').

Although the conflicting claims are not identical, they are not patentably distinct from each other because of corresponding language that recites virtually all of the same elements and functions claimed in the previously patented invention, e.g., "", etc.

The claimed differences would be obvious to a programmer of ordinary skill because the instant claims are merely broader and/or alternate variations of the claims recited in the parent application.

For example, independent claim 1 of the instant application more broadly and/or alternately claims:

"A computing system including operating system software configurable for controlling different computer hardware, comprising:

- (a) a processor;
- (b) at least one storage device;

Art Unit: 2127

- (c) a software operating system operable in a plurality of different computer hardware configurations, the software operating system having modifiable system initialization information stored in the at least one storage device; and
- (d) a system enabler containing information for configuring the software operating system for a computer hardware configuration.

In contrast, claim 1 of the parent application (U.S. Patent '685' more narrowly and/or alternately claims:

"A computing system comprising: **one of** various types of **processors** for executing software; and **a software operating system** for use by said processor, the operating system comprising a boot-up file for beginning execution of an initial portion of a boot-up routine which initial portion of said boot-up routine identifies the type of processor present and passes execution of the boot-up routine; and a self-contained enabler file, **containing processor-specific information**, which receives **execution of the boot-up routine from said operating system and enables said operating system to execute application programs in the identified one of various types of processors using said processor-specific information**, said enabler file being initially **stored in** a read-write **memory device** so that said enabler file may be replaced with an updated enabler file **when system changes are made in said computing system.**"[*Emphasis added to show identical elements*].

Because the instant claims merely eliminate and/or alternately claim limitations from the set of elements and functions claimed in the parent application, such modifications would be readily apparent to a programmer of ordinary skill.

#### **Terminal Disclaimer**

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent is shown to be commonly owned with this application. See 37 CFR 1.130(b). Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

For post GATT applications, (i.e., applications filed after June 8, 1995); the rule § 1.321 (4) (c) (3) requires a provision that must be included. The following requirement is UNCHANGED by GATT and therefore a terminal disclaimer is required for the instant application, i.e., "*shall be enforceable only for and during such period that said patent is commonly owned with the application or patent which formed the basis for the rejection.*"

Art Unit: 2127

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

5. Claims 1-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Sherer et al. (U.S. Pate. No., 5,459,854) in view of Arnold et al. (U.S. Pate. No. 5,128,995).

Per claims 1, and 10, Sherer teaches:

A computing system including software configurable for controlling different computer hardware (Col. 7, lines 52-62, *As mentioned above, a given host **architecture** may or may not include a **configuration** memory, or the **configuration** memory may not include enough information upon which the host **architecture** can be unambiguously identified. Thus, the part of the **initialization** process, as illustrated in FIG. 7, involves executing tests to determine the host environment. In the first system, two aspects of the environment are automatically determined. The CPU type of the host machine, and whether or not the device driver can perform virtual to physical address translations under the OS/2 operating system*), comprising:  
(a) a processor (Fig. 1, B);

(b) at least one storage device (col. 6, lines 51-58, *The **initialization** process for this preferred embodiment is illustrated in FIG. 7. It begins when the operating system reads the program image of the device driver from a secondary **storage** device and loads it into memory. The operating system then branches to the **initialization** code of the program, InitCodeSeg, which has been loaded into memory*);

(c) a software system operable in a plurality of different computer hardware configurations, the software system having modifiable system initialization information stored in the at least one storage device, and regarding modifying the system software see col. 5, lines 55-65, *In the **initialization** module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During **initialization**, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or discarded when the device driver returns from **initialization***); and

(d) a system enabler containing information for configuring the software operating system for a computer hardware configuration (Col. 7, lines 52-62, and lines 66-68, continued on col. 8, lines 1-14, *As mentioned above, a given host **architecture** may or may not include a **configuration** memory, or the **configuration** memory may not include enough information upon which the host*

Art Unit: 2127

*architecture can be unambiguously identified. Thus, the part of the initialization process, as illustrated in FIG. 7, involves executing tests to determine the host environment. In the first system, two aspects of the environment are automatically determined. The CPU type of the host machine, and whether or not the device driver can perform virtual to physical address translations under the OS/2 operating system).*

The reference of Sherer teaches of software enabling software for a wide variety of variant host architecture, but fails to teach of software being **operating system software**. However, the system of Arnold teaches of need for a system initialization tool test that test the BIOS compatibility [or perhaps when the BIOS is corrupted] and load the BIOS from a fixed diskette in case there is no compatibility between the Hardware BIOS of the operating system and the architecture (See, col. 8, lines 38-57, *Previous to the present invention, ROM 36 could include all of the BIOS code which interfaced the operating system to the hardware peripherals. According to one aspect of the present invention, however, ROM 36 is adapted to store only a portion of BIOS. This portion, when executed by the system processor 26, inputs from either the fixed disk 62 or diskette 66 a second or remaining portion of BIOS, hereinafter also referred to as a BIOS image. This BIOS image supersedes the first BIOS portion and being an integral part of the system is resident in main memory such as RAM 32. The first portion of BIOS (ROM-BIOS) as stored in ROM 36 will be explained generally with respect to FIGS. 3-4 and in detail with respect to FIGS. 6A-D. The second portion of BIOS (BIOS image) will be explained with respect to FIG. 5, and the loading of the BIOS image with respect to FIG. 7. Another benefit from loading a BIOS image from a DASD is the ability to load BIOS directly into the system processor's RAM 32. Since accessing RAM is much faster than accessing ROM, a significant improvement in the processing speed of the computer system is achieved. An additional advantage is also gained by storing system utilities on the DASD. When a condition for the usage of the system utilities is required, the system utility can automatically be referenced on the DASD*). However, Arnold boot the BIOS from a diskette image, see col. 3, lines 65-68 continued on col. 4, lines 1-13 (*The protected region of the storage device includes a master boot record, a BIOS image and the system reference diskette image. The BIOS image includes a section known as Power on Self Test (POST). POST is used to test and initialize a system. Upon detecting any configuration error, system utilities from the system reference diskette image, such as set configuration programs, diagnostic programs and utility programs can be automatically activated*). It would have been desirable to put modifiable system initialization information on segment of the storage device as suggested by Sherer, for convenience and usability in case the operating system configuration is not compatible with the computer hardware configuration. For the reason that, the designer who is intent on distributing software in executable form which must occupy a portion of the user accessible space in a computer system, must account for the variant architectures for which the particular device is intended to be used (See Sherer, col. 2, lines 48-52. Therefore, it would have been obvious for a person ordinary skill in the art at the time the invention was made to use operating system software in the invention of Sherer.

Per claim 8, a method for modifying a generic software system to control a plurality of computer hardware systems (see Sherer, col. 5, lines 55-65, *In the **initialization** module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is*

Art Unit: 2127

written for a 286 environment, one for a 386 environments, and so on. During **initialization**, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or discarded when the device driver returns from **initialization**), comprising the steps of:

(a) storing a software operating system and a computer hardware system enabler on a storage device (Col. 7, lines 52-62, and lines 66-68, continued on col. 8, lines 1-14, *As mentioned above, a given host architecture may or may not include a **configuration** memory, or the **configuration** memory may not include enough information upon which the host architecture can be unambiguously identified. Thus, the part of the initialization process, as illustrated in FIG. 7, involves executing tests to determine the host environment. In the first system, two aspects of the environment are automatically determined. The CPU type of the host machine, and whether or not the device driver can perform virtual to physical address translations under the OS/2 operating system, also see col. 5, lines 55-65, In the **initialization** module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During **initialization**, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or discarded when the device driver returns from **initialization**), regarding the storage device and the operating system, see Sherer (col. 6, lines 51-58, *The **initialization** process for this preferred embodiment is illustrated in FIG. 7. It begins when the operating system reads the program image of the device driver from a secondary storage device and loads it into memory. The operating system then branches to the **initialization** code of the program, InitCodeSeg, which has been loaded into memory*);*

(b) transferring the software operating system and system enabler from the storage device (See Sherer, col. 5, lines 55-65, *In the **initialization** module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During **initialization**, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment*); and

(c) modifying the software operating system, with information from the system enabler file, to adapt the software operating system for operation on a computer hardware system (see Sherer, col. 5, lines 55-65, *In the **initialization** module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During **initialization**, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or*

Art Unit: 2127

*discarded when the device driver returns from initialization*). Regarding the modifying software operating system instead of software see the obviousness reasons stated in the rejection of claim 1.

Per claims 2-3, wherein the system enabler is stored in a nonvolatile read write memory storage device, and wherein the system enabler is stored in a read only memory are both suggested by Arnold in col. 7, lines 1-14 (*The local bus 28 is further connected through a bus controller 34 to a read only memory (ROM) 36 on the planar 24. An additional nonvolatile memory (NVRAM) 58 is connected to the microprocessor 26 through a serial/parallel port interface 40, which is further connected to bus controller 34. The nonvolatile memory can be CMOS with battery backup to retain information whenever power is removed from the system. Since the ROM is normally resident on the planar, model and submodel values stored in ROM are used to identify the system processor and the system planar I/O configuration respectively. Thus these values will physically identify the processor and planar I/O configuration*), for the reason to be able to have O.S. software and different patches for different architecture in one place when the system is booted up.

Per claim 4, wherein the system enabler includes selectable software patches and resources, the system of Sherer teaches of selectable software in col.5, lines 55-65 (*In the initialization module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During initialization, the device driver determines its environment and selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or discarded when the device driver returns from initialization*), for the reason to be bale to select appropriate patches when the initialization is not compatible and therefore makes the system operational in case of incompatibility of with the system architecture.

Per claim 5, including processor means for transferring the software operating system and system enabler from a storage device to a random access memory, the system of Arnold teaches of the limitation in col. 8, lines 18-37, and col. 9, lines 16-47 (*In general, a first program such as ROM-BIOS prechecks the system and loads a BIOS master boot record into RAM. The master boot record includes*). The Examiner emphasizes that in the system of Arnold the selected BIOS is offloaded from a diskette, however, for the obviousness reasons stated above, the system of Sherer has a RAM selector segment which select the segment of the RAM that include the selected compatible software for loading of the selected initialization segments (See Sherer, col. 17, lines 53-62 of the code)

Per claim 6, including a plurality of system enablers containing date and hardware compatibility information, the data compatibility information is taught by Sherer in, col. 5, lines 55-65 (*In the initialization module, several different versions of the TransmitChain procedure are stored in a TransmitChain segment. One version is written for a 286 environment, one for a 386 environments, and so on. During initialization, the device driver determines its environment and*

Art Unit: 2127

*selects the appropriate TransmitChain procedure for that environment. The selected procedure is then moved to the beginning of the TransmitChain segment. The rest of the TransmitChain procedures which were not appropriate for this environment are overwritten or discarded when the device driver returns from initialization).*

Per claim 7 The computing system according to claim 6, wherein the software operating system utilizes a particular system enabler, is taught by Sherer in col. 5, lines 48-63 (*During initialization, the device driver determines its environment and selects the **appropriate** TransmitChain procedure for that environment*).

Per claim 9, and the limitation of (a) storing a plurality of system enablers containing computer hardware compatibility information and selection criteria in computer system nonvolatile read-write memory, See the rejection of claims 2 and 3 above; and (b) selecting from said plurality of system enablers a system enabler 'file having compatible information corresponding to a computer hardware configuration, see the rejection of claim 7 above.

Per claims 11-12, see the rejection of claims 2-3 above

Per claim 13, see the rejection of claim 5 above.

Per claim 14, please see the rejection of claim 5 above

Per claim 15, please see the rejection of claim 6 above.

Per claims 16, and 18, wherein the software operating system utilizes the system enabler with a most recent date-time stamp, it is notoriously well known to include the date and time of the software version as a date-time stamp, for the reason to be able to identify different versions from each other.

Per claim 17, wherein the system enabler contains information corresponding to a machine state, the information containing machine state is the information related to specific architecture and it is taught by Sherer in col. 1, lines 61-68, continued on col. 2, lines 1-10 (*A manufacturer of add-on devices designed to handle a plurality of variant architecture must provide software for controlling the add-on devices which runs in each of the variant architectures. This software is typically referred to as a device driver. Therefore, typically prior art systems required a number of versions of device drivers written for each variant architecture for which the add-on device is intended to run. The user must be able to select which device driver applies to his data processing system, and the manufacturer must distribute a large number of versions. This requires an intelligent user for loading and initializing the device driver.*).

Per claim 19, wherein the system enabler contains information corresponding to selection criteria, the limitation is taught by Sherer in col. 11, lines 16-22 (*The third word 148c in the*

Art Unit: 2127

*entry 148 of the bounds table is an offset from the beginning of the functional segment code group to the first byte beyond the end of the code block 149. Thus, the second and third words provide the bounds of the code block. This information is used when moving the code block into lower memory during a relocation process).*

Per claim 20, the method of claim 19 wherein the software operating system utilizes the system enabler with a most appropriate selection criteria (See, Sherer in col. 11, lines 23-31, *The address table 143 contains a list of location dependent entries in the resident data segment that must be modified so that they refer to the appropriate locations in a relocated code block. For example, the entry in the MAC upper dispatch table that contains the address of the TransmitChain routine must be updated to point to the new location of the relocated TransmitChain. Every code block has associated with it an address table*).

### **Conclusion**

6. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to **Majid A. Banankhah** whose voice telephone number is (703) 308-6903. A voice mail service is also available at this number.

All response sent to U.S. Mail should be mailed to:

**Commissioner of Patent and Trademarks  
Washington, D.C. 20231**

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Majid Banankhah

3/18/04

  
**MAJID BANANKHAH  
PRIMARY EXAMINER**